
Proceedings Paper:
Calinescu, RADU CONSTANTIN, Bassett, Micah, DEVLIN-HILL, BRENDAN et al. (2026) A Tool for the Verification and Synthesis of Stochastic World Models. In: 38th International Conference on Computer Aided Verification 2026. Lecture Notes in Computer Science. Springer.

A Tool for the Verification and Synthesis of Stochastic World Models

Radu Calinescu, Micah Bassett, Brendan Devlin-Hill, Simos Gerasimou, Sinem Getir Yaman, Kavan Fatehi, and Gricel Vázquez

Department of Computer Science, University of York, UK

Abstract. We present a tool for the compositional verification and correct-by-construction synthesis of *stochastic world models*—heterogeneous networks of interdependent stochastic models including discrete and continuous-time Markov chains, Markov decision processes (MDPs), partially observable MDPs, and stochastic multi-player games. Through its unique integration of multiple probabilistic and parametric model checking paradigms, our tool unifies the modelling, verification and synthesis of systems characterised by a combination of probabilistic and nondeterministic uncertainty, discrete and continuous-time behaviour, partial observability, and multi-agent interaction.

1 Introduction

Many systems must satisfy strict safety, reliability and performance requirements while operating under significant uncertainty arising from the nondeterminism, probabilistic behaviours, and partial observability inherent, for instance, in their inputs, randomised actions, and imperfect environment perception. Probabilistic model checking (PMC) [3, 49] is an established formal approach for analysing these key quantitative requirements. To this end, PMC uses (i) stochastic models such as discrete and continuous-time Markov chains (DTMCs and CTMCs), Markov decision processes (MDPs), partially observable MDPs (POMDPs), or stochastic multiplayer games (SMGs) to model the relevant behaviour of the systems under verification, and (ii) expressive probabilistic temporal logics such a probabilistic computation tree logic (PCTL) [32] and continuous stochastic logic (CSL) [1] to specify a wide range of properties over these models.

PMC and the probabilistic model checkers that implement its techniques (e.g., PRISM [41] and Storm [33]) have been highly successful [39, 42, 45] when applied to systems whose behaviour can be expressed using one of these model types. However, a growing number of systems comprise components and/or present aspects characterised by a combination of probabilistic and nondeterministic uncertainty, discrete and continuous-time behaviour, partial observability, and multi-agent interaction. As such, verifying these systems requires the joint analysis of multiple types of interdependent models with properties expressed in different probabilistic temporal logics. The tool for the compositional verification and correct-by-construction synthesis of heterogeneous networks of interdependent stochastic models or *stochastic world models* (SWMs) presented

in our paper, is designed to perform such analyses, which existing PMC techniques and tools cannot handle. Developed by our project *'Universal Stochastic Modelling, Verification & Synthesis Framework (ULTIMATE)'*, this tool (also called 'ULTIMATE') is freely available online [62, 63] together with ample usage instructions and a suite of case studies designed to support its adoption.

A preliminary tool version was briefly mentioned when we introduced its underlying verification theory in [6]. Here we present for the first time the fully-fledged tool, which incorporates many significant advancements over [6]:

1. *Synthesis of stochastic world models*—One of the two core functionalities of the tool, the synthesis capability, is introduced for the first time.
2. *Enhanced verification & analytics*—The GUI now supports systematic execution of series of verifications that vary model parameters, with new interactive 2D/3D plotting and exporting of verification results.
3. *Command-line interface (CLI)*—A new CLI enables the tool's integration into larger verification pipelines, offering a headless alternative to the GUI.
4. *Performance improvements & lazy evaluation*—A major refactoring of the internal data model enables lazy evaluation/advanced caching, with verification subtasks performed only when necessary, and results efficiently reused.
5. *Usability Improvements*—The tool was augmented with real-time progress indicators, improved text selection, dynamic model management, etc.

Beyond these new features, this paper provides the first comprehensive description of the tool's architecture, implementation and scalability, alongside the first set of case studies demonstrating its use to synthesise stochastic world models.

2 SWM Verification and Synthesis Problems

Stochastic world models. The SWMs that our tool operates with are finite sets of interdependent parametric stochastic models[1] (DTMCs, CTMCs, MDPs, POMDPs and SMGs) specified in the high-level modelling language of the probabilistic model checker PRISM [41]. Three types of parameters may be used in the stochastic models of an SWM handled by our tool:

1. *Dependency parameters*, which capture inter-model relationships where the transition probabilities/rewards of one model depend on a property of another model. This property corresponds to a performance, dependability or another attribute of the system component associated with the second model.
2. *External parameters*, which represent observable but uncontrollable environmental factors or system characteristics. Their values are derived by applying statistical inference to data from sources such as pre-deployment testing, historical logs, runtime monitoring or domain-expert estimates.
3. *Internal parameters*, which correspond to controllable decision variables or configuration options of the modelled system. They can be Boolean, discrete or continuous, supporting the modelling of binary decisions, selection between multiple options, or fine-grained configuration, respectively.

[1] A *parametric stochastic model* [16, 21, 35, 39] is a stochastic model with transition probabilities/rates and rewards defined as rational functions over a set of parameters.

Formally, a stochastic world model is a tuple $\mathcal{U} = (M, D, E, I)$, where:

- M is a set of PMC-supported stochastic state-transition models such that the transition probabilities/rates and rewards of each model $m \in M$ are defined over mutually disjoint sets $D(m)$ of *dependency parameters*, $E(m)$ of *external parameters*, and $I(m)$ of *internal parameters*.
- D is a set of *inter-model dependencies* (m, d, m', ϕ) such that:
  (i) $(m, d, m', \phi) \in D$ if and only if $m \in M$, $d \in D(m)$, and the value of d is to be *calculated* by applying PMC to property ϕ of model $m' \in M \setminus \{m\}$;
  (ii) for every model $m \in M$ and every parameter $d \in D(m)$, there exists exactly one $(m, d, m', \phi) \in D$.
- E is a set of *external-parameter definitions* (m, e, f, O) such that:
  (i) $(m, e, f, O) \in E$ if and only if $m \in M$, $e \in E(m)$, and the value of e is to be *estimated* by applying the inference function f to the *observations* (e.g., log entries and/or runtime measurements) O;
  (ii) for every model $m \in M$ and every parameter $e \in E(m)$, there exists exactly one $(m, e, f, O) \in E$.
- I is a set of *internal-parameter definitions* (m, x, X) such that:
  (i) $(m, x, X) \in I$ if and only if $m \in M$, $x \in I(m)$, and the value of x is to be *selected* from the set of possible values X as part of the development or runtime adaptation of the system modelled by $\mathcal{U}$;
  (ii) for every model $m \in M$ and every parameter $x \in I(m)$, there exists exactly one $(m, x, X) \in I$.

SWM verification problem. Given a stochastic world model with no internal parameters (i.e., $\forall m \in M : I(m) = \{\}$), and a formally specified property ϕ_v of a model $m_v \in M$, the verification problem tackled by our tool is to establish the value of property ϕ_v, i.e., to compute $pmc(m_v, \phi_v)$. Solving this verification problem in the most general case involves:

- the PMC of additional models from M to evaluate the dependency parameters of m_v, which is particularly complex in the presence of co-dependent models;
- the synthesis of suitable policies for any verified models (e.g., MDPs and POMDPs) that contain nondeterminism;
- the estimation of the external parameters of all models involved (directly or through model interdependencies) in the verification.

The novel verification algorithm used by our tool [6] performs these steps through a recursive depth-first search analysis of the strongly connected component (SCC) tree induced by the stochastic world model, with each SCC analysed through a combination of *parametric model checking*[2] and numerical methods.

[2] Parametric model checking extends the applicability of PMC to certain types of parametric stochastic models and properties (e.g., unbounded reachability in DTMCs and CTMCs, but not bounded CTMC reachability). Its application yields rational expressions over the parameters of the verified model.

SWM synthesis problem. Given a stochastic world model with internal parameters (i.e., $\exists m \in M : I(m) \neq \{\}$), and

- a set of $n_c \geq 0$ constraints $pmc(m_j, \phi_j) \in \Phi_j$, where $m_j \in M$, ϕ_j is a formally specified property of m_j, and Φ_j is the set of allowed values for that property;
- a set of $n_o \geq 1$ optimisation objectives '$\mathsf{maximise}\ pmc(m_k, \phi_k)$' or '$\mathsf{minimise}\ pmc(m_k, \phi_k)$', where $m_k \in M$, and ϕ_k is a formally specified property of m_k,

Our tool uses multi-objective genetic algorithms (MOGAs) adapted from [5,25, 27] to synthesise sets of internal parameter valuations which (i) ensure that the n_c constraints are satisfied and (ii) yield stochastic world model instances that are Pareto-optimal with respect to the n_o optimisation objectives.

3 Related work

To the best of our knowledge, no other tools support the formal verification and synthesis of heterogeneous sets of interdependent stochastic models with the characteristics summarised in Section 2. A theoretical foundation for the compositional assume-guarantee verification of interdependent probabilistic automata has been developed in [11,22,46,50], but it is not implemented by existing probabilistic model checking tools. Furthermore, its theoretical results do not cover heterogeneous sets of models, external parameters, and synthesis—all of which are key features of our tool and its underpinning theory, and are required for the case studies described in Section 6.

Specifically, PRISM and Storm use the technique to decompose large monolithic stochastic models into SCCs that can be analysed compositionally. In contrast, our tool applies SCC decomposition to the dependency graph induced by the verified SWM, to determine the verification order of its component models. Finally, while synthesis of stochastic models is supported by tools such as Prophesy [18], EvoChecker [25,26] and RODES [4], these tools only operate with single stochastic models.

4 Architecture and Implementation

As illustrated in Fig. 1, the operation of our tool is managed by a Verification & Synthesis Orchestrator, which can be invoked through:

1. A Java graphical user interface (GUI) that assists the user with (i) assembling SWMs (by selecting predefined stochastic models from the user's computer, and organising them into a SWM through defining their parameters and interdependencies), and (ii) specifying verification or synthesis problems.
2. A Java command-line interface that allows the Orchestrator to solve verification and synthesis problems in headless mode. This enables automated pipeline integration via the command '`java -jar ultimate-headless.jar -pf` *`SWM`* `-m` *`model`* `-p` *`prop`* `-o` *`output-dir`*', where the arguments specify the SWM definition (i.e, *project*) file, the verification target (an ID of a model from the SWM, and either a property or a file containing a list of properties), and the results directory, respectively.

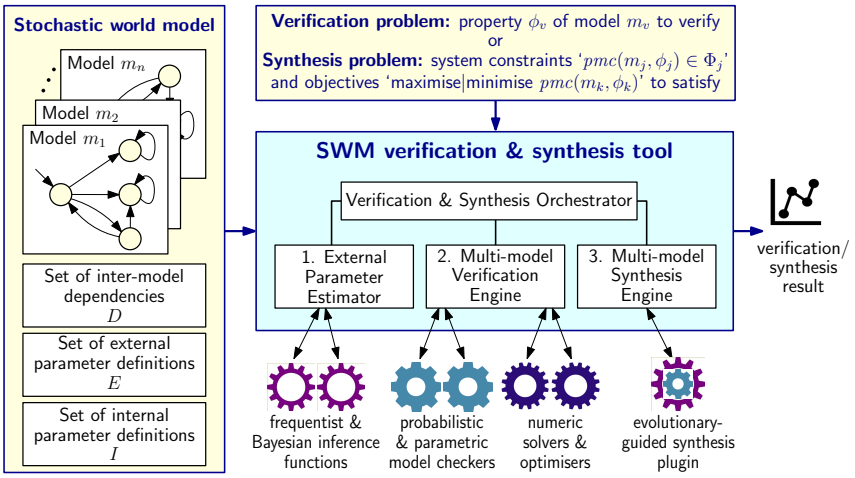

Fig. 1. Tool architecture

The Orchestrator performs the following tasks:

1. It leverages an External Parameter Estimator, which uses a suite of predefined frequentist and Bayesian inference functions (including our implementations of the KAMI Bayesian estimator [19, 29] and of simple mean functions) to estimate the external parameters of the SWM models.
2. It solves verification problems using a Multi-model Verification Engine, which implements the SWM verification algorithm summarised in Section 2 and detailed in [6] (and in Appendix D.2). This involves invoking (i) the probabilistic model checkers PRISM [41], PRISM-games [48] and Storm [33] (with a user option for selecting a preferred model checker) via their APIs, and (ii) numeric solvers and optimisers from the Scipy library [59] to handle circular dependencies in conjunction with the parametric model checking functions of PRISM/Storm.
3. It solves synthesis problems through a Multi-model Synthesis Engine that utilizes a MOGA evolutionary-guided synthesis plugin. This plugin is based on EvoChecker, our single stochastic model synthesis approach [5,25,27] (see Appendix E for details).

Additionally, the Orchestrator manages systematic verification experiments across external parameter ranges; when triggered via the GUI, the results of these experiments can be rendered as graphs to facilitate system analysis. Finally, the tool employs lazy evaluation, ensuring external parameters are only estimated and properties are only verified when absolutely needed. It also caches these estimates and verification results for efficient reuse whenever possible.

5 Using the SWM Verification & Synthesis Tool

To illustrate the tool's capabilities, we use it to verify the stochastic world model of a Robotic Assistive Dressing (RAD) cyber-physical system; this SWM is inspired by our recent work in [58]. The RAD system consists of four components:

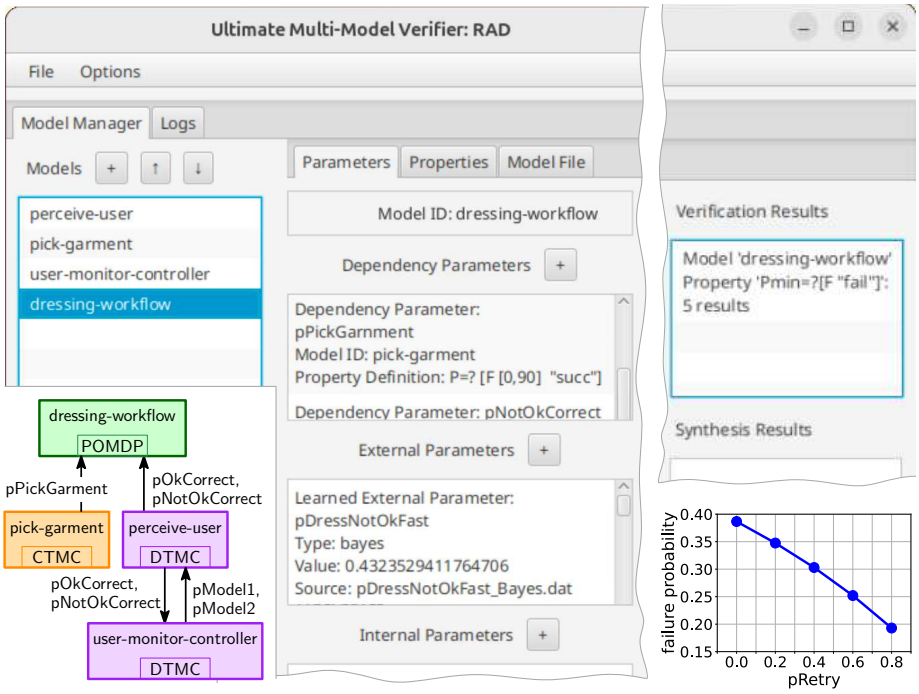

Fig. 2. Verification of the RAD stochastic world model in ULTIMATE

Robot arm: picks garment used for dressing from a peg and slides its sleeves onto a disabled user's arms; modelled by a pick-garment CTMC used to establish the probability pPickGarment of successful garment picking within 90s by verifying CSL property $P_{=?}[F^{[0,90]}$ "success"] subject to external parameter pRetry (probability of retrying the garment picking after a failed attempt).

User monitor: uses deep-learning perception to determine the user's state (ok/not ok); modelled by a perceive-user DTMC whose analysis yields pOkCorrect and pNotOkCorrect, the probabilities of correctly predicting the user state when the user is "ok" and "not ok", respectively.

Low-level controller: configures user monitor; modelled by a user-monitor-controller DTMC whose analysis gives probabilities pModel1 and pModel2 with which different perception models are utilized by the user monitor.

High-level controller: drives the dressing process execution; modelled by a dressing-workflow POMDP that accounts for the true user state being unobservable, and has dependency parameters pPickGarment, pOkCorrect and pNotOkCorrect.

Fig. 2 shows the four models' dependencies (bottom-left), the models added to a *verification project* for the RAD system (top-left), the specification of several parameters of the dressing-workflow POMDP (middle), alongside with the verification results (bottom-right) for the PCTL property $Pmin_{=?}[F$ "fail"] (the minimum probability that the dressing fails, top-right) of this POMDP across several values of the external parameter pRetry from the pick-garment CTMC.

6 Case Studies and Experimental Results

Our tool comes with a repository of case studies [63] that demonstrate its application across a wide range of domains, stochastic world models, and verification and synthesis problems. We describe a subset of these case studies below.

Verification case studies. In addition to the RAD case study from the previous section, we summarise four further verification case studies whose stochastic world model graphs and selected verification results are shown in Fig. 3.

Smart Motion Detection (SMD) verifies a system with a motion sensor and a smart light, modelled as codependent DTMCs. The sensor's detection probability depends on the lighting level, which in turn is determined by the sensor's output. The verification results establish how the probability pLargeObject of a large object being detected and the smart lighting's power use vary with the probability of a large object being present in the monitored area.

Dynamic Power Management for Foreign eXchange system (DPM-FX) verifies a service-based system (modelled as a DTMC taken from [27]) that uses a database deployed on a hard disk controlled by a dynamic power manager (modelled as a CTMC adapted from [12]). The verification results establish how the expected FX execution time varies across different predefined values of the external DPM parameter pIdle2Sleep (the probability that the hard disk is put to sleep when idle).

Mobile Robot Fleet (RoboFleet) verifies a fleet of N robots (modelled as MDPs) managed by a human supervisor (modelled as a DTMC). The verification results establish how the expected number of failed robots and the supervisor-intervention cost vary when the external parameter nAttempts (maximum number of attempts to unstuck blocked robots) is varied for a two-robot system.

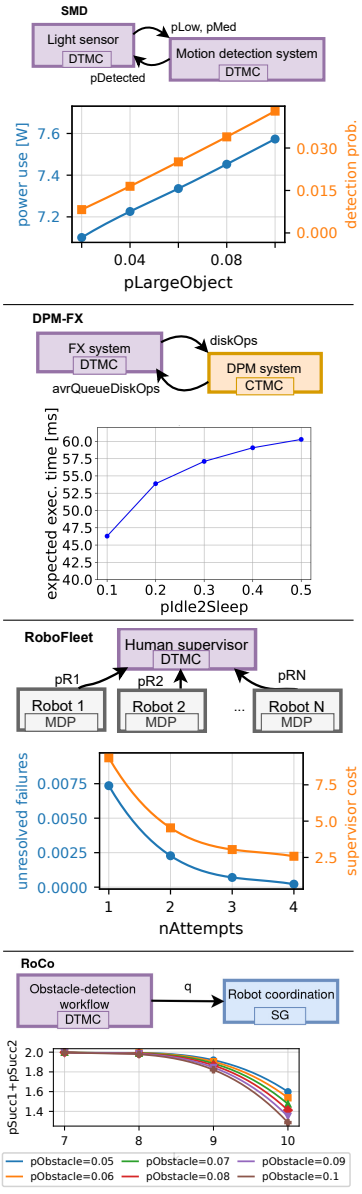

Fig. 3. Verification case studies

Robot Coordination (RoCo) verifies two robots coordinating to navigate a grid world while avoiding collisions. The robots are modelled as an SMG (adopted from [43]), and their movement is influenced by a camera-based obstacle-detection workflow (modelled as a DTMC); this determines the probability q of

robots moving in a direction adjacent to the planned one. The verification results establishe how the combined success probability pSucc1+pSucc2 for both robots to reach their targets within k moves varies with the probability pObstacle of a grid cell containing an obstacle.

Synthesis case studies. We also summarise three SWM synthesis case studies from [63]. Fig. 4 shows their SWM graphs and synthesised Pareto fronts.

SmartFarming uses a stochastic world model with internal parameters to synthesise solutions that are Pareto-optimal with respect to maximising the success probability and minimising the cost of a vineyard maintenance mission inspired by our recent work on human-robot teaming [65]. The mission is performed by two mobile robots (modelled as MDPs) that use low/high-risk collision detection components (modelled as DTMCs) and operate under the control of a planner (modelled as a DTMC) that manages the tasks of both the robots and farm workers.

RAD-synth uses a variant of the RAD stochastic world model from Section 5. The low-level controller is removed, and the parameters pModel1, pModel2, and pRetry are redefined as internal parameters that require synthesis. The tool generates parameter valuations that ensure a garment-picking success probability of at least 0.9 within 90s, while achieving optimal trade-offs among three objectives: minimising robot energy consumption, and minimising the dressing failure probability under both best-case and worst-case execution scenarios.

Casino uses an SWM to synthesise solutions that are Pareto-optimal with respect to maximising a casino cheater's winnings and probability of evading detection, with both the cheater and the casino modelled as DTMCs. This synthesis determines the optimal strategy given the trade-off between the two objectives.

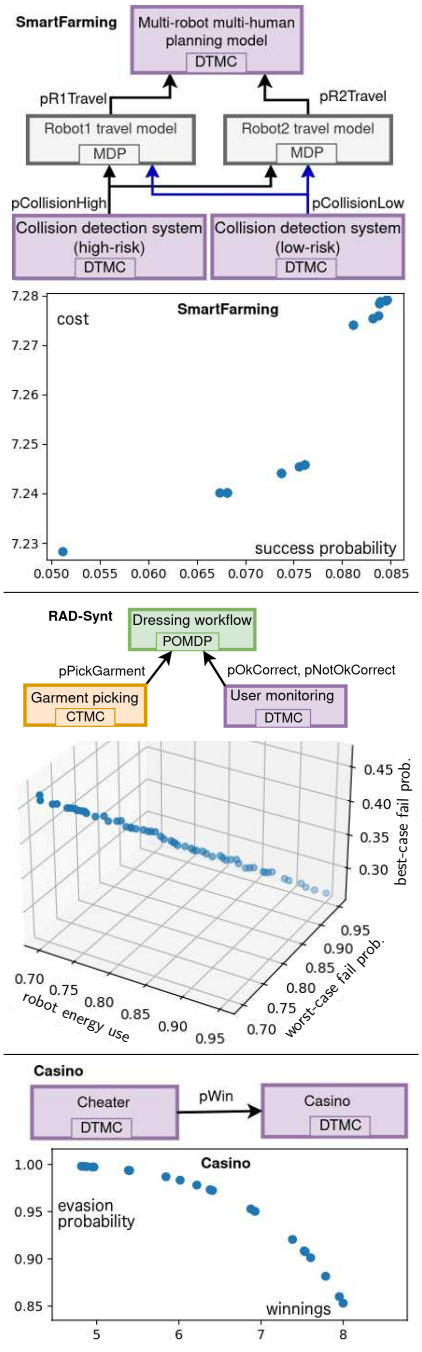

Fig. 4. Synthesis case studies

7 Scalability

Complexity analysis sketch. We provide a high-level characterisation of the tool's computational overhead here, with a detailed analysis available in Appendix D.3 and Appendix E.3. Verifying a SWM with no circular dependencies requires probabilistic model checking for: (i) each property defining one of its $\#D$ dependency parameters (at most once, since some dependencies may not be relevant for the property being verified, and thanks to caching), and (ii) the original top-level property. When circular dependencies exist, the properties (i) of models within the SWM's strongly connected components are established using:

1. parametric model checking, with the resulting system of polynomial equations solved numerically via Newton-Raphson's optimisation method [53] from the Python library SciPy (`https://scipy.org`), when the model/property types requiring analysis are supported by parametric model checking;
2. Powell's optimisation method [56] from SciPy, performing probabilistic model checking for each property during every method iteration, otherwise.

These computations are dominated by the model checking tasks, which are required at most $\#D+1$ times to verify properties (i) and (ii) in the former case, or at most $N_{\mathsf{Powell}} \cdot \#D+1$ times when using Powell's method over N_{Powell} iterations.

Stochastic world model synthesis involves verifying a SWM property for each of the n_c constraints and n_o optimisation objectives of the SWM synthesis problem, for every individual from the population P evolved by the MOGA, and for each of the N_{MOGA} iterations of the evolution. This results in $(n_c+n_o)\cdot\#P\cdot N_{\mathsf{MOGA}}$ total SWM verifications, each incurring the complexity discussed above.

While SWM verification and synthesis require numerous model checking operations, scalability is maintained through three key factors:

1. Linear scaling—The number of such operations is linear with respect to the number of dependencies ($\#D$), Powell iterations (N_{Powell}), synthesis requirements ($n_c + n_o$), and MOGA population size ($\#P$) and iterations (N_{MOGA}).
2. High-performance back-ends—Verification is performed by PRISM, PRISM-games and Storm, which are state-of-the-art tools optimised for high-efficiency probabilistic/parametric model checking [33, 34, 41, 48].
3. Compositionality—A key advantage of the tool is its analysis of individual component models rather than a monolithic state space. While a monolithic model[3] would suffer from a state-space explosion (with states totalling up to the product of all component state sizes), our tool keeps the verification overhead manageable by exploiting the inherent modularity of the SWM.

Experimental evaluation. As summarised in Table 1, all individual verification results for the case studies from Section 6 were obtained in under 6s. The synthesis tasks that generated the Pareto fronts shown in Fig. 4 were completed in between 14–39 minutes (for a MOGA population of 100 solutions over 10 iterations) on a virtual machine (VM) with the specification from the table. While these models are relatively small, a study of Markov chains in leading software engineering (SE) venues that we carried out [61] shows that such model sizes are representative of the abstractions commonly used at least in SE research.

[3] This is hypothetical, since the heterogeneous models of SWMs cannot be combined.

Table 1. ULTIMATE verification (top) and synthesis (bottom) case studies

ID	Domain/System*	Models	States†	Deps‡	Time+
RAD [58]	Assistive-care/CPS	CTMC, DTMCx2, POMDP	52	7	3.10s
SMD	Security/IoT	DTMCx2	13	3	5.30s
DPM-FX	Finance/SBS	DTMC [27], CTMC [12]	66	2	1.50s
RoboFleet	Logistics/Multi-robot	DTMC, MDPx2	43	2	0.06s
RoCo	Patrolling/Dual-robot	DTMC, SMG [43]	9816	1	14.2s
SmartFarming [65]	Farming/CPS-Human	MDPx2, DTMCx3	1208	6	39min
RAD-synth [58]	Assistive-care/CPS	CTMC, DTMC, POMDP	48	3	27min
Casino	Gaming/Protocol	DTMCx2	15	1	14min

*CPS=cyber-physical system, IoT=Internet of Things, SBS=service-based system
†total number of states across all SWM models
‡total number of SWM dependency parameters
+verification time: mean over 10 executions on M1 MacBook Pro with 32GB RAM; synthesis time: 100 chromosomes×10 MOGA iterations on VirtualBox-7 VM with 2 vCPUS & 8GB RAM

Table 2. Scalability results

Model	SWM1	SWM2	SWM3
small	0.79s	2.06s	12.26s
medium	1.22s	3.65s	22.74s
large	9.57s	35.14s	233.36s

To assess scalability for larger state spaces, we conducted two sets of experiments (models available at [63]). For the first experiment, we verified the property $P_{=?}[F\ \mathsf{success}]$ for variants of the FX model from our DPM-FX case study across three model sizes: 283 states (*small*), 41,873 states (*medium*), and 529,239 states (*large*). For each of these, we verified SWMs with increasing complexity: SWM1 (3 FX models, 1 dependency level, 3 dependencies), SWM2 (11 FX models, 2 dependency levels, 10 dependencies), and SWM3 (75 FX models, 3 dependency levels, 74 dependencies). The results, obtained on a MacBook Pro (M1, 32GB RAM), are shown in Table 2. Notably, for the largest SWM comprising over $39 \cdot 10^6$ states (75 large models), the verification was completed in under 4 minutes.

For the second experiment, we assembled a synthetic SWM by combining three of the largest models from the widely used PRISM benchmark suite [44]: crowds with TotalRuns=6, CrowdSize=20 (a 10,633,591-state DTMC), cluster for N=512 (a 9,465,876-state CTMC), and wlan6 (a 5,007,552-state MDP). We introduced dependencies by setting two crowds probabilities as parameters dependent on reachability properties of wlan6 and cluster. Our tool required 324s to verify a top-level crowds property (on a VM with the specification from Table 1), with 319s (98.4%) of that time spent on PRISM invocations for verification subtasks. This confirms that our tool effectively supports state-of-the-art benchmark model sizes with negligible overhead over that of the back-end model checkers.

8 Conclusion

We presented the tool 'ULTIMATE' for verifying and synthesising heterogeneous sets of interdependent stochastic models. We detailed its architecture and demonstrated its versatility through case studies whose stochastic world models included DTMCs, CTMCs, MDPs, POMDPs and/or SMGs, complemented by scalability experiments. Future extensions will include support for incremental verification [37, 47] and verification with confidence-intervals [7, 9]. Additionally, we aim to speed-up the resolution of model co-dependencies by integrating recent advances in compositional parametric model checking [20, 21, 51].

References

1. Aziz, A., Sanwal, K., Singhal, V., Brayton, R.: Verifying continuous time Markov chains. In: Computer Aided Verification. pp. 269–276. Springer (1996)
2. Baier, C., Haverkort, B., Hermanns, H., Katoen, J.P.: Model checking continuous-time Markov chains by transient analysis. In: Emerson, E.A., Sistla, A.P. (eds.) Computer Aided Verification. pp. 358–372. Springer Berlin Heidelberg, Berlin, Heidelberg (2000)
3. Baier, C., Katoen, J.: Principles of model checking. MIT Press (2008)
4. Calinescu, R., Ceska, M., Gerasimou, S., Kwiatkowska, M., Paoletti, N.: RODES: A robust-design synthesis tool for probabilistic systems. In: Bertrand, N., Bortolussi, L. (eds.) 14th International Conference on Quantitative Evaluation of Systems (QEST). Lecture Notes in Computer Science, vol. 10503, pp. 304–308. Springer (2017). https://doi.org/10.1007/978-3-319-66335-7_20
5. Calinescu, R., Ceska, M., Gerasimou, S., Kwiatkowska, M., Paoletti, N.: Efficient synthesis of robust models for stochastic systems. J. Syst. Softw. **143**, 140–158 (2018). https://doi.org/10.1016/J.JSS.2018.05.013
6. Calinescu, R., Getir Yaman, S., Gerasimou, S., Vázquez, G., Bassett, M.: Verification of multi-model stochastic systems. In: 48th IEEE/ACM International Conference on Software Engineering (ICSE 2026) (2026), to appear. Preprint available at https://drive.google.com/file/d/1icMGEs1cx92BAeRUiJ4-2nV06Y3aunUX/view
7. Calinescu, R., Ghezzi, C., Johnson, K., Pezzè, M., et al.: Formal verification with confidence intervals to establish quality of service properties of software systems. IEEE Trans. Reliab. **65**(1), 107–125 (2016). https://doi.org/10.1109/TR.2015.2452931
8. Calinescu, R., Imrie, C., Mangal, R., et al.: Controller synthesis for autonomous systems with deep-learning perception components. IEEE Trans. Software Eng. pp. 1–22 (2024). https://doi.org/10.1109/TSE.2024.3385378
9. Calinescu, R., Johnson, K., Paterson, C.: FACT: A probabilistic model checker for formal verification with confidence intervals. In: Chechik, M., Raskin, J. (eds.) 22nd International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS). Lecture Notes in Computer Science, vol. 9636, pp. 540–546. Springer (2016). https://doi.org/10.1007/978-3-662-49674-9_32
10. Calinescu, R., Johnson, K., Rafiq, Y.: Using observation ageing to improve Markovian model learning in QoS engineering. In: 2nd ACM/SPEC International Conference on Performance engineering. pp. 505–510 (2011)
11. Calinescu, R., Kikuchi, S., Johnson, K.: Compositional reverification of probabilistic safety properties for large-scale complex it systems. In: Monterey Workshop. pp. 303–329. Springer (2012)
12. Calinescu, R., Kwiatkowska, M.Z.: Using quantitative analysis to implement autonomic IT systems. In: 31st International Conference on Software Engineering (ICSE). pp. 100–110. IEEE (2009). https://doi.org/10.1109/ICSE.2009.5070512
13. Calinescu, R., Paterson, C., Johnson, K.: Efficient parametric model checking using domain knowledge. IEEE Trans. Software Eng. **47**(6), 1114–1133 (2021). https://doi.org/10.1109/TSE.2019.2912958
14. Calinescu, R., Rafiq, Y., Johnson, K., Bakır, M.E.: Adaptive model learning for continual verification of non-functional properties. In: 5th ACM/SPEC International Conference on Performance engineering. pp. 87–98 (2014)

15. Chen, T., Forejt, V., Kwiatkowska, M., Parker, D., Simaitis, A.: Automatic verification of competitive stochastic systems. Formal Methods in System Design **43**, 61–92 (2013)
16. Daws, C.: Symbolic and parametric model checking of discrete-time Markov chains. In: First International Conference on Theoretical Aspects of Computing (ICTAC). pp. 280–294 (2005)
17. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002)
18. Dehnert, C., Junges, S., Jansen, N., Corzilius, F., Volk, M., Bruintjes, H., Katoen, J.P., Ábrahám, E.: Prophesy: A probabilistic parameter synthesis tool. In: Computer Aided Verification: 27th International Conference, CAV 2015, San Francisco, CA, USA, July 18-24, 2015, Proceedings, Part I 27. pp. 214–231. Springer (2015)
19. Epifani, I., Ghezzi, C., Mirandola, R., Tamburrelli, G.: Model evolution by run-time parameter adaptation. In: 31st International Conference on Software Engineering. pp. 111–121. IEEE (2009)
20. Fang, X., Calinescu, R., Gerasimou, S., Alhwikem, F.: Fast parametric model checking through model fragmentation. In: 43rd IEEE/ACM International Conference on Software Engineering (ICSE). pp. 835–846. IEEE (2021). https://doi.org/10.1109/ICSE43902.2021.00081
21. Fang, X., Calinescu, R., Gerasimou, S., Alhwikem, F.: Fast parametric model checking with applications to software performability analysis. IEEE Trans. Software Eng. **49**(10), 4707–4730 (2023). https://doi.org/10.1109/TSE.2023.3313645
22. Feng, L.: On learning assumptions for compositional verification of probabilistic systems. Ph.D. thesis, University of Oxford (2014)
23. Forejt, V., Kwiatkowska, M., Norman, G., Parker, D.: Automated verification techniques for probabilistic systems. Formal Methods for Eternal Networked Software Systems: 11th International School on Formal Methods for the Design of Computer, Communication and Software Systems, SFM 2011, Bertinoro, Italy, June 13-18, 2011. Advanced Lectures 11 pp. 53–113 (2011)
24. Gainer, P., Hahn, E.M., Schewe, S.: Accelerated model checking of parametric Markov chains. In: International Symposium on Automated Technology for Verification and Analysis (ATVA). pp. 300–316. Springer (2018)
25. Gerasimou, S., Calinescu, R., Tamburrelli, G.: Synthesis of probabilistic models for quality-of-service software engineering. Autom. Softw. Eng. **25**(4), 785–831 (2018). https://doi.org/10.1007/S10515-018-0235-8
26. Gerasimou, S., Cámara, J., Calinescu, R., et al.: Evolutionary-guided synthesis of verified Pareto-optimal MDP policies. In: 36th IEEE/ACM International Conference on Automated Software Engineering (ASE). pp. 842–853. IEEE (2021). https://doi.org/10.1109/ASE51524.2021.9678727
27. Gerasimou, S., Tamburrelli, G., Calinescu, R.: Search-based synthesis of probabilistic models for quality-of-service software engineering (T). In: Cohen, M.B., Grunske, L., Whalen, M. (eds.) 30th IEEE/ACM International Conference on Automated Software Engineering (ASE). pp. 319–330. IEEE Computer Society (2015). https://doi.org/10.1109/ASE.2015.22
28. Gerasimou, S., Tamburrelli, G., Calinescu, R.: Search-based synthesis of probabilistic models for quality-of-service software engineering (T). In: Cohen, M.B., Grunske, L., Whalen, M. (eds.) 30th IEEE/ACM International Conference on Automated Software Engineering (ASE). pp. 319–330. IEEE Computer Society (2015). https://doi.org/10.1109/ASE.2015.22

29. Ghezzi, C., Tamburrelli, G.: Predicting performance properties for open systems with KAMI. In: Mirandola, R., Gorton, I., Hofmeister, C. (eds.) Architectures for Adaptive Software Systems. pp. 70–85. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
30. Hahn, E.M., Han, T., Zhang, L.: Synthesis for PCTL in parametric markov decision processes. In: NASA formal methods symposium (NFM). pp. 146–161. Springer (2011)
31. Hahn, E.M., Hermanns, H., Wachter, B., Zhang, L.: PARAM: A model checker for parametric Markov models. In: International Conference on Computer Aided Verification (CAV). pp. 660–664 (2010)
32. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects of Computing **6**(5), 512–535 (1994)
33. Hensel, C., Junges, S., Katoen, J.P., Quatmann, T., Volk, M.: The probabilistic model checker Storm. International Journal on Software Tools for Technology Transfer pp. 1–22 (2022)
34. Hensel, C., Junges, S., Quatmann, T., Volk, M.: Riding the storm in a probabilistic model checking landscape. In: Jansen, N., Junges, S., Kaminski, B.L., Matheja, C., Noll, T., Quatmann, T., Stoelinga, M., Volk, M. (eds.) Principles of Verification: Cycling the Probabilistic Landscape : Essays Dedicated to Joost-Pieter Katoen on the Occasion of His 60th Birthday, Part II, pp. 98–114. Springer Nature Switzerland (2025). https://doi.org/10.1007/978-3-031-75775-4_5
35. Jansen, N., Corzilius, F., Volk, M., Wimmer, R., Ábrahám, E., Katoen, J.P., Becker, B.: Accelerating parametric probabilistic verification. In: International Conference on Quantitative Evaluation of Systems (QEST). pp. 404–420 (2014)
36. Jansen, N., Junges, S., Katoen, J.P.: Parameter synthesis in Markov models: A gentle survey. Principles of Systems Design: Essays Dedicated to Thomas A. Henzinger on the Occasion of His 60th Birthday pp. 407–437 (2022)
37. Johnson, K., Calinescu, R., Kikuchi, S.: An incremental verification framework for component-based software systems. In: 16th International ACM Sigsoft Symposium on Component-based Software Engineering (CBSE). pp. 33–42 (2013)
38. Junges, S., Ábrahám, E., Hensel, C., Jansen, N., Katoen, J.P., Quatmann, T., Volk, M.: Parameter synthesis for Markov models: covering the parameter space. Formal Methods in System Design pp. 1–79 (2024)
39. Katoen, J.P.: The probabilistic model checking landscape. In: 31st Annual ACM/IEEE Symposium on Logic in Computer Science. pp. 31–45 (2016)
40. Kemeny, J.G., Snell, J.L., Knapp, A.W.: Denumerable Markov Chains, 2nd edition, Graduate Texts in Marhematics, vol. 40. Springer (1976)
41. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) 23rd International Conference on Computer Aided Verification (CAV). LNCS, vol. 6806, pp. 585–591. Springer (2011)
42. Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic model checking: Advances and applications. In: Formal System Verification. pp. 73–121. Springer (2017)
43. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: Equilibria-based probabilistic model checking for concurrent stochastic games. In: Proc. 23rd International Symposium on Formal Methods (FM'19). LNCS, vol. 11800, pp. 298–315. Springer (2019)
44. Kwiatkowska, M., Norman, G., Parker, D.: The prism benchmark suite. In: 9th International Conference on Quantitative Evaluation of SysTems. pp. 203–204. IEEE CS press (2012)

45. Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic model checking and autonomy. Annual Review of Control, Robotics, and Autonomous Systems **5**, 385–410 (2022)
46. Kwiatkowska, M., Norman, G., Parker, D., Qu, H.: Assume-guarantee verification for probabilistic systems. In: 16th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS). pp. 23–37. Springer (2010)
47. Kwiatkowska, M., Parker, D., Qu, H.: Incremental quantitative verification for markov decision processes. pp. 359 – 370 (07 2011). https://doi.org/10.1109/DSN.2011.5958249
48. Kwiatkowska, M., Parker, D., Wiltsche, C.: PRISM-games: Verification and strategy synthesis for stochastic multi-player games with multiple objectives. Int. J. Softw. Tools Technol. Transf. **20**(2), 195–210 (2018). https://doi.org/10.1007/s10009-017-0476-z
49. Kwiatkowska, M.Z., Norman, G., Parker, D.: Stochastic model checking. In: Formal Methods for Performance Evaluation, 7th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM. LNCS, vol. 4486, pp. 220–270. Springer (2007)
50. Liu, Y., Li, R.: Compositional stochastic model checking probabilistic automata via assume-guarantee reasoning. International Journal of Networked and Distributed Computing **8**(2), 94–107 (2020)
51. Mertens, H., Quatmann, T., Katoen, J.P.: Compositional reasoning for parametric probabilistic automata. In: 36th International Conference on Concurrency Theory (CONCUR 2025). pp. 31–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik (2025)
52. Nebro, A.J., Durillo, J.J., Luna, F., Dorronsoro, B., Alba, E.: MOCell: A cellular genetic algorithm for multiobjective optimization. International Journal of Intelligent Systems **24**(7), 726–746 (2009)
53. Nocedal, J., Wright, S.J.: Numerical optimization. Springer (1999)
54. Norman, G., Parker, D., Zou, X.: Verification and control of partially observable probabilistic systems. Real-Time Systems **53**(3), 354–402 (2017)
55. Organization, W.H.: Ageing and health (2021), https://www.who.int/news-room/fact-sheets/detail/ageing-and-health, accessed: 2024-12-05
56. Powell, M.J.: A view of algorithms for optimization without derivatives. Mathematics Today-Bulletin of the Institute of Mathematics and its Applications **43**(5), 170–174 (2007)
57. Quatmann, T., Dehnert, C., Jansen, N., Junges, S., Katoen, J.P.: Parameter synthesis for Markov models: faster than ever. In: 14th International Symposium on Automated Technology for Verification and Analysis (ATVA). pp. 50–67. Springer (2016)
58. Rafiq, Y., Vázquez, G., Calinescu, R., Dogramadzi, S., Hierons, R.M.: Symbolic runtime verification and adaptive decision-making for robot-assisted dressing. In: Taibi, D., Smite, D. (eds.) Software Engineering and Advanced Applications. pp. 290–308. Springer Nature Switzerland, Cham (2026)
59. SciPy: Python optimization libray (2025), https://scipy.org/
60. Tarjan, R.E.: Depth-first search and linear graph algorithms. SIAM J. Comput. **1**, 146–160 (1972), https://api.semanticscholar.org/CorpusID:16467262
61. TASP Research Group – University of York: Discrete-event Markov chains used in 2016-2020 research papers published in leading software engineering venues (2021), https://www.cs.york.ac.uk/tasp/VERACITY/Resources/DTMCs-from-papers-in-leading-SE-venues-2016-2020.pdf

62. ULTIMATE: Integrated platform for the verification and synthesis of stochastic world models (2025), https://doi.org/10.5281/zenodo.17332288
63. ULTIMATE project: Github repository (2025), https://github.com/ULTIMATE-YORK/ULTIMATE
64. United Nations Population Fund (UNFPA): The state of world population 2022: Seeing the invisible (2022), https://www.unfpa.org/ageing, accessed: 2024-12-05
65. Vázquez, G., Evangelidis, A., Shahbeigi, S., Gerasimou, S.: Adaptive human-robot collaborative missions using hybrid task planning. In: 2025 IEEE/ACM 20th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). pp. 73–84 (2025). https://doi.org/10.1109/SEAMS66627.2025.00016
66. Vázquez, G., Evangelidis, A., Shahbeigi, S., Gerasimou, S.: Adaptive human-robot collaborative missions using hybrid task planning. In: 2025 IEEE/ACM 20th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). pp. 73–84. IEEE (2025)
67. Zhao, X., Gerasimou, S., Calinescu, R., et al.: Bayesian learning for the robust verification of autonomous robots. Nature Comms. Eng. **3**(1), 18 (2024)
68. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength pareto evolutionary algorithm. TIK report **103** (2001)